REMARKS

Applicants are in receipt of the Office Action mailed March 25, 2004. Claims 1-110 remain pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Section 103(a) Rejection:

The Office Action rejected claims 1-12, 14-16, 18-26, 28-30, 32-40, 42-50, 52-54, 56-64, 66-68, 70-78, 80-86, 88-90, 92-97, 99-101, 103-108 and 110 under 35 U.S.C. § 103(a) as being unpatentable over Holiday (U.S. Patent 6,421,739) in view of Meth et al. (U.S. Patent 6,401,216) (hereinafter "Meth").

Regarding claim 1, Applicants disagree with the Examiner's contention that Holiday in view of Meth teaches a method comprising: executing a process within the virtual machine, wherein the virtual machine comprises a virtual machine virtual memory manager, wherein said executing the process comprises: the process referencing an object in a virtual heap during execution, wherein the virtual heap comprises an in-memory heap and a store heap, the process accessing the referenced object in the in-memory heap and checkpointing a state of the process executing on the virtual machine to a first memory space, wherein said checkpointing comprises the virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap.

Applicants assert that Holiday in view of Meth fails to teach wherein the virtual machine comprises a virtual machine virtual memory manager, wherein the virtual heap comprises an in-memory heap and a store heap, and wherein said checkpointing comprises the virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap.

Holiday teaches a method for providing fault tolerance to a Java Virtual Machine (JVM) by checkpointing objects that are created, modified, and/or deleted during the

process of responding to an event of a transaction. Holiday further teaches that the checkpointed objects are sent to, and stored in, a second JVM such that the second JVM is fully capable of continuing the processing of the transaction in the event of the failure of the first JVM. (Holiday, column 2, lines 16-28).

However, Holiday fails to teach wherein the virtual machine comprises a virtual machine virtual memory manager. In contrast, Holiday teaches a Java Virtual Machine that includes a heap memory, a checkpoint environment, and garbage collection functions (Holiday, column 2, line 66 – column 3, line 2). Holiday also teaches that a JVM may also include a system interface, an execution engine, and threads (Holiday, column 4, lines 21-29). Holiday fails to teach the use of a virtual machine virtual memory manager.

Further, applicants also assert that Holiday fails to teach a process referencing an object in a virtual heap during execution, wherein the virtual heap comprises an in-memory heap and a store heap. There is no virtual heap in Holiday. In fact, Holiday refers only to heap memory and applications residing in memory (Holiday, Figures 1, 3-18, column 2, line 66-column3, line 2, column 3, lines 39-40). Applicants can find no reference in Holiday to a virtual heap or to any of the functions or practices associated with a virtual heap. Additionally, Holiday teaches that "garbage collection functions 36 and 46 are responsible for managing heap memory and freeing up heap memory occupied by data objects that are no longer referenced by the running applications." (Holiday, column 3, lines 9-26). Nowhere does Holiday mention that his memory heap includes both an in-memory heap and a store heap.

The passages cited by the Examiner (Heap Memories 32 and 42, column 3, lines 1-67, and column 10, lines 1-16) in support of his contentions regarding the teaching of Holiday refer only to normal (non-virtual) heap and memory functions, such as garbage collecting (Heap Memories 32 and 42, and column 3, lines 9-38), software applications receiving and processing events (column 3, lines 39-67), and the storing and restoring of checkpoint data (column 10, lines 1-16). Applicants can find no reference in the cited passages that pertain to virtual machine virtual memory managers, virtual heaps

comprising in-memory heaps and store heaps, or flushing sections of an in-memory heap to a store heap.

Applicants also submit that Holiday also fails to teach wherein said checkpointing comprises the virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap. Since, as shown above, Holiday fails to teach a virtual machine virtual memory manager, and also fails to teach a virtual heap comprising an in-memory heap and a store heap, Holiday must also clearly and necessarily fail to teach a virtual machine virtual memory manager flushing one or more sections of an in-memory heap to a store heap.

Applicants can find no relevance in the Examiner's cited passages (Figure 2 - step 224, column 5, lines 65-67, and column 6, lines 1-64), to a virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap. For example, step 224 of Figure 2 only mentions generating a checkpoint, but says nothing regarding flushing portions of an in-memory heap to a store heap. Further, column 5, lines 65-67 state, "[i]n step 224, the content of the checkpoint is calculated, and the write barrier discontinues tracking changes to the heap space (for the purpose of checkpointing)." Column 6, lines 1-64 go on to describe various aspects of checkpointing and sending the checkpoint data to another JVM through the use of RMI. Applicants note, however, that none of these passages teach or suggest anything regarding a virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap.

Meth teaches a system for checkpointing parallel programs including writing message data and signal states to a checkpoint file and for copying the message data and signal states from the checkpoint file to memory to restore the program. Each process is responsible for taking its own checkpoint as scheduled by a coordinating process. (Meth, Abstract).

The Examiner argues that Meth teaches that if the referenced object is in the store heap and not in the in-memory heap when referenced by the process, the virtual machine virtual memory manager copying a section of the store heap comprising the referenced object from the store heap to the in-memory heap. Applicants disagree with the Examiner's interpretation of Meth. The Examiner's cited passage of Meth (column 6, lines 16-43) refer to the normal loading of processing into the memory space of a computer system and specifically describes the memory of such a system as including programming code, global variables, a heap and a stack. Applicants fail to see the relevance of this portion of Meth. Applicants can find no mention in either the cited passage, nor anywhere else in Meth, that teaches or suggests a virtual machine virtual memory manager copying a section of a store heap comprising a referenced object to an in-memory heap. In fact, applicants can find no reference in Meth to any virtual machine, virtual machine virtual memory manager, or a heap comprising a store heap and an in-memory heap.

Thus, applicants can find no reference in Holiday or Meth, either separately or in combination, that teaches or suggests a method comprising executing a process within the virtual machine, wherein the virtual machine comprises a virtual machine virtual memory manager, wherein said executing the process comprises: the process referencing an object in a virtual heap during execution, wherein the virtual heap comprises an in-memory heap and a store heap, wherein, if the referenced object is in the store heap and not in the in-memory heap when referenced by the process, the virtual machine virtual memory manager copying a section of the store heap comprising the referenced object from the store heap to the in-memory heap; the process accessing the referenced object in the in-memory heap and checkpointing a state of the process executing on the virtual machine to a first memory space, wherein said checkpointing comprises the virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap.

Applicants also disagree with the examiner's statement that, "[i]t would have been obvious to one of ordinary skill in the art at the time the invention was made to combine

the teachings of Meth and Holiday because the teaching of Meth would improve the system of Holiday by executing a process. Applicants can also find no suggestion or motivation to combine Holiday and Meth in the Examiner's cited passage (Meth, column 6, lines 16-17) which states, "[e]ach process of a parallel program is loaded in the memory of the computing unit that is to execute the process." This passage refers to the fact, under Meth, individual processes that make up a parallel program are loaded into the memory of a computer system. Further, Applicants fail to find any expressed or implied benefit to combine Meth with Holiday in the Examiner's cited passage. In fact, Applicants can find no reference or teaching in Holiday or Meth, either singularly or in combination that suggests any motivation or benefit to such a combination.

Applicants also remind the examiner that "[t]he art must fairly teach or suggest to one to make the specific combination as claimed. That one achieves an improved result by making such a combination is no more than hindsight without an initial suggestion to make the combination." *In re Dembiczak*, 175 F.3d 994, 50 USPQ2d 1614 (Fed. Cir. 1999); see also M.P.E.P. § 2143.01. Thus, applicants assert that examiner argument, "[i]t would have been obvious ... to combine the teachings of Meth and Holiday because the teaching of Meth would improve the system of Holiday" does not establish a proper basis for an obviousness rejection.

Thus, in light of the above remarks, applicants assert that the rejection of claim 1 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 1 apply to claim 81.

Regarding claim 15, applicants assert that Holiday fails to teach a method comprising: expiring one or more leases to services for the first process on the virtual machine. The Examiner cites column 3, lines 10 – 26 and refer to Holiday's Garbage Collection Functions 36/46. The Examiner's cited references describe the garbage collection functions of Holiday that "are responsible for managing heap memory and freeing up heap memory occupied by data objects that are no longer referenced by the running application." (Holiday, column 3, lines 10-13). Applicants assert that garbage

collection, even the generation-copying garbage collection functions described by Holiday, does not have anything to do with leases to services for a process. Garbage collection is well known in the art and is a completely separate concept from a lease to a service. Applicants further submit that even if the garbage collection functions of Holiday represent a lease to a service, although applicants assert that they do not, Holiday still does not teach the expiring of leases. In fact, applicants can find no reference in Holiday or Meth, either alone or in combination, regarding leases to services or the expiring of leases to services for a process on a virtual machine.

Further regarding claim 15, the examiner admits that Holiday does not teach stopping the process execution on the virtual machine, reading the stored state of the process from the persistent store, reconstituting the stored state of the process on the virtual machine, establishing the one or more leases to services for the process on the virtual machine and resuming the process execution on the virtual machine. The Examiner relies upon Meth to teach this and cites two passages (column 8, lines 44-67, and column 10, lines 47-61) to support his assertion that Meth teaches stopping the process execution on the virtual machine. In addition, the Examiner refers to Figure 9 and step 1006 of figure 10. Applicants submit that nowhere in the cited passages or in the figures does Meth describe stopping the process execution on the virtual machine. Figure 9 displays a flowchart that described several steps, none of which include stopping process execution on the virtual machine. Step 1106 of Figure 10 states, "save needed variables from current environment." The text supporting and describing Figure 9, and 10, discuss Meth's method for checkpointing and restarting processes, but neither discuss stopping process execution on the virtual machine. Meth describes how each process is responsible for taking its own checkpoint (Meth, Abstract, column 3, lines 18-28, column 6, lines 49-58) and also teaches that when restarting parallel programs, "each participating user process of the parallel program initiates a restart process" (Meth, column 10, lines 34-36). Applicants submit that the processes of a parallel program cannot initiate a restart process, by calling mp_restart for example (Meth, column 10, lines 36-39), if process execution on the virtual machine has been stopped. Thus, Meth teaches away from stopping process execution on the virtual machine. Further,

applicants can find no teaching in Holiday that teaches stopping process execution on the virtual machine. Thus, neither Holiday nor Meth, nor the combination of Holiday and Meth, teach the stopping of process execution on the virtual machine.

Applicants also assert that, contrary to the Examiner's contention, Meth fails to disclose establishing the one or more leases to services for the process on the virtual machine. Applicants can find no such reference in Meth, or Holiday, that teaches or suggests establishing leases to services for a process on a virtual machine. Firstly, as shown above regarding claim 1, Meth fail to teach a virtual machine and therefore must also fail to teach establishing leases to services for a process on a virtual machine. Secondly, Meth fails to teach anything at all regarding leases in any way. The passages cited by the Examiner (Meth, Steps 1008 – 1016 of Figure 10, column 10, lines 62-67, and column 11, lines 1-65), do not refer to establishing leases to services as the examiner claims. Instead the cited steps illustrated in Figure 10 refer to reading and restoring data, file offsets, and signal states of a process. Applicants fail to see the relevance of the Examiner's second cited passage (column 10, lines 62-67) that refers to determining whether a program that is restarting is the same program that was running when the checkpoint information was saved. Column 11, lines 1, 65, describe, according to Meth, the reading and restoring of checkpoint information to a restarting process. Applicants submit, however that simply restoring checkpoint information is not related to establishing leases to services. Leases to services are considered external to a process and are not typically considered part of the process state. It is not known or suggested in the prior art to re-establish a previous state of leases when reconstituting the state of a process on a virtual machine. The Examiner has not shown any reference that suggests this and applicants can find no such reference or suggestion in Meth or Holiday, either alone or in combination.

Thus, applicants assert that, as shown above, Holiday in view of Meth fails to teach all of the limitations of claim 15 and therefore the Examiner has not established a proper case of obviousness. Therefore, in light of the above remarks, applicants assert that the rejection of claim 15 is not supported by the cited art and withdrawal of the

rejection is respectfully requested. Similar remarks as discussed above in regard to claim 15 apply to claims 53 and 89.

Regarding claim 29, Applicants disagree with the Examiner's assertion and submit that Holiday fails to teach expiring one or more leases to services for the first process on the virtual machine. As shown above regarding claim 15, Holiday fails to teach expiring leases to services for a process on a virtual machine. The Examiner's cited references describe the garbage collection functions of Holiday that "are responsible for managing heap memory and freeing up heap memory occupied by data objects that are no longer referenced by the running application." (Holiday, column 3, lines 10-13). Applicants assert that garbage collection, even the generation-copying garbage collection functions described by Holiday, does not have anything to do with leases to services for a process. Garbage collection is well known in the art and is a completely separate concept from a lease to a service. Applicants further submit that even if the garbage collection functions of Holiday are a lease to a service, although applicants assert that they are not, Holiday still does not teach the expiring of leases. In fact, applicants can find no reference in Holiday or Meth regarding leases to services or the expiring of leases to services for a process on a virtual machine.

The Examiner admits that holiday fails to teach suspending the first process executing within the virtual machine, reading the state of a suspended second process from the persistent store, wherein the state of the second process was stored to the persistent store prior to said executing the first process within the virtual machine, reconstituting the one or more leases to services for the second process on the virtual machine and resuming the execution of the second process within the virtual machine. The Examiner relies upon Meth to teach these limitations that Holiday fails to teach. However, Meth, in fact, fails to teach suspending the first process on executing within the virtual machine. The Examiner's cited references, including step 900 of Figure 9, and column 8, lines 44-67, describe the process of checkpointing as taught by Meth. Under Meth, message traffic is stopped, signals are blocked, and file offsets and signal states are saved. Applicants assert that stopping message traffic, blocking signals and saving file

offsets or signal states are very different from suspending the first process executing within the virtual machine. In fact, since Meth teaches that all processes participate in both the checkpointing and restarting processes (Meth, Abstract, column 3, lines 18-28, column 6, lines 49-58, column 10, lines 34-36), such processes cannot be suspended during these processes. Thus, Meth actually teaches away from suspending a process executing within a virtual machine. Additionally, applicants can find no teaching or suggestion in either Holiday or Meth, either alone or in combination, that teach suspending a process executing within a virtual memory.

Further regarding claim 29, applicants submit that, as shown regarding claim 15, Meth does not disclose establishing one or more leases to services for the second process on the virtual machine. Please refer to applicants' arguments regarding claim 15 for specific reasons whey Meth fails to teach establishing leases to services for a process on a virtual machine. Applicants can find no teaching or suggestion in either Holiday or Meth, either alone or in combination, regarding establishing leases to services for a process on a virtual machine.

Thus, applicants assert that, as shown above, Holiday in view of Meth fails to teach all of the limitations of claim 29 and therefore the Examiner has not established a proper case of obviousness. In light of the above remarks, applicants assert that the rejection of claim 29 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 29 apply to claims 67 and 100.

Regarding claim 43, Holiday fails to teach a system wherein a first memory is configured to store a store heap for the first process, wherein the store heap is comprised within a virtual heap for the first process. As shown above regarding claim 1, Holiday fails to teach a virtual machine virtual memory manager and therefore must also fail to teach a virtual heap. In fact, Holiday refers only to heap memory and applications residing in memory (Holiday, Figures 1, 3-18, column 2, line 66-column3, line 2, column 3, lines 39-40). Applicants can find no reference in Holiday to a virtual heap or to any of

the functions or practices normally associated with virtual heaps. The Examiner's cited references (JVM 40 of Figure 2, column 5, lines 1-67, and column 6, lines 1-64) refer to event processing, garbage collection and various aspects of checkpointing and sending the checkpoint data to another JVM through the use of RMI. Applicants can find no reference in the cited passages regarding a virtual heap. Further, applicants can find so such reference or suggestion in either Holiday or Meth, either alone or in combination.

Additionally, Holiday also fails to teach wherein the in-memory heap comprises cached portions of the store heap for access by the first process. The examiner admits that Holiday does not explicitly teach that the in-memory heap comprises cached portions of the store heap for access by the first process, but does contend that "cache portion/line are inherent in a heap." Applicants disagree. "In relying upon the theory of inherency, the examiner must provide a basis in fact and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic necessarily flows from the teachings of the applied prior art." (underlining in the original) (M.P.E.P. § 2112, paragraph 5). Heaps do not inherently comprise an in-cached portions of a store heap for access by a process. Store heaps are not inherent in heaps. Holiday makes no reference to any sort of cache or caching. Further, applicants can find no reference in Holiday or Meth, either alone or in combination, that teaches an in-memory heap comprising cached portions of a store heap for access by a process.

Further regarding claim 43, applicants assert that, contrary to the Examiner's statement, Holiday does not teach a virtual machine virtual heap manager configured to flush one or more sections of the in-memory heap to the store heap. Firstly, the Examiner admits that Holiday does not teach a device configured to perform operations on the virtual heap according to a virtual machine virtual heap manager. Thus, applicants assert that if Holiday fails to teach a virtual machine virtual heap manager, Holiday must necessarily also fail to teach a virtual machine virtual heap manager configured to flush one or more sections of an in-memory heap to a store heap. The Examiner's cited references (steps 224-230 of Figure 2, column 5, lines 65-67, and column 6, lines 1-56) refer to event processing, garbage collection and various aspects of checkpointing and

sending the checkpoint data to another JVM through the use of RMI. Applicants can find no reference in the cited passage regarding a virtual machine virtual heap manager as the Examiner contends. Furthermore, applicants can find no such teaching or suggestion in Holiday or Meth, either alone or in combination.

The Examiner relies upon Meth to disclose a device configured to perform operations on the virtual heap according to a virtual machine virtual heap manager, wherein the virtual machine virtual heap manger is configured to copy a section of the store heap comprising an object from the store heap to the in-memory heap in response to the first process referencing the object in the virtual heap when the referenced object is in the store heap and not in the in-memory heap. Applicants disagree. The examiner supports his erroneous contention by citing the loading of each process of a parallel program into the memory of a computing unit that is to execute the program (Meth, column 6, lines 16-43). Applicants submit that Meth is describing only the normal process of loading an executable process into memory and regarding the normal code and data sections of process memory, but does not describe anything, in the Examiner's cited passages or elsewhere, regarding a virtual machine virtual heap manager copying a section of a store heap comprising an object to the in-memory heap in response to a process referencing the object when the object is in the store heap and not in the in-memory heap. The normal loading of an executable process in to the memory of a computer is different and distinct from a virtual machine virtual heap manager copying sections of a store heap to an in-memory heap in response to a process accessing an object in the store heap when that object is not in the in-memory heap. Further, applicants can find no such teaching or suggestion in Meth or Holiday, either alone or in combination.

Applicants also remind the examiner that "[t]he art must fairly teach or suggest to one to make the specific combination as claimed. That one achieves an improved result by making such a combination is no more than hindsight without an initial suggestion to make the combination." *In re Dembiczak*, 175 F.3d 994, 50 USPQ2d 1614 (Fed. Cir. 1999); see also M.P.E.P. § 2143.01. Thus, applicants assert that examiner argument, "[i]t

would have been obvious ... to combine the teachings of Meth and Holiday because the teaching of Meth would improve the system of Holiday" does not establish proper basis for an obviousness rejection.

Thus, in light of the above remarks, applicants assert that the rejection of claim 43 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

The Office Action rejected claims 13, 51 and 87 under 35 U.S.C. § 103(a) as being unpatentable over Holiday in view of Meth, and further in view of Gall (U.S. Patent 6,480,862). These claims are patentable for at least the reasons given above in regard to their respective independent claims.

The Office Action rejected claims 17, 31, 55, 69, 91 and 102 under 35 U.S.C. § 103(a) as being unpatentable over Holiday in view of Meth, and further in view of Shirakihara, et al. (U.S. Patent 5,802,267) (hereinafter "Shirakihara"). These claims are patentable for at least the reasons given above in regard to their respective independent claims.

The Office Action rejected claims 27, 41, 65, 79, 98 and 109 under 35 U.S.C. § 103(a) as being unpatentable over Holiday in view of Meth, and further in view of Stiffler, et al. (U.S. Patent 6,622,263) (hereinafter "Stiffler"). These claims are patentable for at least the reasons given above in regard to their respective independent claims.

Applicants further note that the rejection is improper because the Examiner has not shown that Stiffler qualifies as a prior art reference. The Examiner has the burden of proof to produce the factual basis for the rejection. *In re Warner*, 154 USPQ 173, 177 (C.C.P.A. 1967), *cert. denied*, 389 U.S. 1057 (1968). Since the Examiner has not proven that Stiffler qualifies as a prior art reference, the Examiner has not met this burden of proof and the rejection is improper. More specifically, the Stiffler patent was filed on June 30, 2000, after Applicants' filing date of June 2, 2000. Stiffler does claim

the benefit of a provisional application filed June 30, 1999. However, the June 30, 1999 filing date can only be used as Stiffler's 35 U.S.C. § 102(e) prior art date for the subject matter that is common to both the Stiffler patent and the provisional application. Since it is common practice for a later filed utility application to include more or different subject matter than its earlier provisional application, it is unclear whether the material in Stiffler relied upon by the Examiner was actually present in Stiffler's provisional application. Therefore, Applicants request that the Examiner provide a copy of Stiffler's provisional application and show that the subject matter on which the Examiner is relying on to reject Applicants' claims is also present in Stiffler's provisional application. Until the Examiner has made this showing, the rejection is improper. *See, In re Wertheim*, 209 USPQ 554 (CCPA 1981).

Moreover, the Stiffler patent is not entitled to the June 30, 1999 date as a section 102(e) prior art date unless at least one claim of the Stiffler patent is supported (under 35 U.S.C. § 112) in the provisional application. Under 35 U.S.C. 119(e)(1), a patent is not entitled to its provisional application's filing date as a prior art date unless at least one claim of the published utility application is supported (per 35 U.S.C. § 112) in the provisional application. The rejection is improper unless the Examiner can show that Stiffler's published application has the necessary claim support in the provisional application to be entitled to the provisional application's filing date as its § 102(e) prior art date. *See also* M.P.E.P. § 2136.03(IV).

Since the Examiner has not provided the necessary evidence to show that the Stiffler patent is prior art to the present application, the current rejection is improper.

Applicants also assert that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

Information Disclosure Statements:

Applicants note that four different information disclosure statements with accompanying Forms PTO-1449 were submitted on August 6, 2001, August 16, 2001, September 17, 2001 and March 17, 2003, respectively. The signed and initialed form PTO-1449 was returned from only the first of these IDSs. Applicants request the Examiner to carefully consider the listed references and return copies of the signed and initialed Forms PTO-1449 from other three IDSs.

CONCLUSION

Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-63200/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

☐ Petition for Extension of Time

☐ Notice of Change of Address

☐ Fee Authorization Form authorizing a deposit account debit in the amount of $ for fees ( ).

☐ Other:

Respectfully submitted,

Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: June 25, 2004